TQS: Product specification report

***Ana Alexandra Antunes [876543]*, *Bruno Bernardes [876543]***

v2024-04-18

[1 Introduction 1](#_Toc39943375)

[1.1 Overview of the project 1](#_Toc39943376)

[1.2 Limitations 1](#_Toc39943377)

[2 Product concept 2](#_Toc39943378)

[2.1 Vision statement 2](#_Toc39943379)

[2.2 Personas 2](#_Toc39943380)

[2.3 Main scenarios 2](#_Toc39943381)

[2.4 Project epics and priorities 2](#_Toc39943382)

[3 Domain model 2](#_Toc39943383)

[4 Architecture notebook 3](#_Toc39943384)

[4.1 Key requirements and constrains 3](#_Toc39943385)

[4.2 Architetural view 3](#_Toc39943386)

[4.3 Deployment architeture 4](#_Toc39943387)

[5 API for developers 4](#_Toc39943388)

[6 References and resources 4](#_Toc39943389)

# Introduction

## Overview of the project

Este projeto tem como objetivo principal aplicar os conhecimentos teóricos e práticos adquiridos para garantir a qualidade de um sistema de reserva e gerenciamento de voos para aeroportos. O cadeira de TQS foca em técnicas para identificar e corrigir defeitos em software, além de promover a qualidade em todas as fases do desenvolvimento.

O nosso produto, denominado "GateMate", é uma solução abrangente projetada para simplificar e aprimorar a experiência de viagem dos utilizadores que utilizam serviços aéreos. A plataforma oferece uma variedade de funcionalidades, incluindo reserva de bilhetes de avião, check-in online e acesso a informações de voos em tempo real.

O " GateMate" é uma ferramenta indispensável tanto para os passageiros quanto para os aeroportos. Para os viajantes, proporciona conveniência e praticidade ao permitir que realizem as suas reservas e check-ins de forma rápida e fácil, além de fornecer informações atualizadas sobre os horários de partida e embarque dos voos. Já para os aeroportos, contribui para a eficiência operacional ao agilizar o processo de check-in e reduzir a necessidade de assistência presencial.

Este oferece também uma oportunidade única para aplicar metodologias de teste, como teste de unidade, teste de integração e teste de sistema, para garantir que o "GateMate" atenda aos mais altos padrões de qualidade. Além disso, será dada ênfase à automação de testes e à análise de métricas de qualidade para monitorar e melhorar continuamente a eficácia do software.

Dessa forma, o projeto "GateMate" não apenas demonstra a aplicação prática dos conceitos aprendidos na cadeira de TQS, mas também oferece uma solução valiosa para melhorar a experiência de viagem dos utilizadores e a eficiência dos serviços aeroportuários.

## Limitations

Até ao momento, o projeto encontra-se em fase inicial de conceção, sem que qualquer funcionalidade tenha sido implementada. No entanto, durante o planejamento inicial, identificamos uma potencial limitação que merece destaque: a integração com as companhias aéreas.

Essa integração é essencial para oferecer aos utilizadores uma ampla gama de opções de reserva de voos, garantindo a abrangência e a utilidade do nosso serviço. No entanto, reconhecemos que a integração com os sistemas de reservas das companhias aéreas pode apresentar desafios significativos. Questões como a complexidade dos sistemas existentes, requisitos técnicos específicos e até mesmo políticas internas das companhias podem impactar o processo de integração.

Portanto, é importante ressaltar que estamos cientes desses possíveis obstáculos e estamos preparados para enfrentá-los de maneira proativa. O nosso objetivo é garantir uma integração eficiente e robusta, proporcionando uma experiência de reserva de voos tranquila e satisfatória para os utilizadores.

# Product concept and requirements

## Vision statement

Este sistema é uma plataforma de reserva e gerenciamento de voos projetada para simplificar e aprimorar a experiência de viagem dos utilizadores que utilizam serviços aéreos. Ele será utilizado pelos viajantes para realizar diversas operações relacionadas a voos, como reserva de bilhetes, check-in online e acesso a informações de voos em tempo real.

O problema de alto nível que o nosso sistema busca resolver é a complexidade e a falta de conveniência frequentemente associadas às viagens aéreas. Atualmente, os viajantes muitas vezes precisam lidar com múltiplos sites e aplicativos de companhias aéreas diferentes para fazer as suas reservas, realizar check-in e obter informações sobre os seus voos. Isso pode ser confuso e demorado, resultando numa experiência de viagem frustrante.

"GateMate" foi concebido para abordar esse problema, oferecendo uma solução abrangente e centralizada para todas as necessidades de reserva e gerenciamento de voos dos utilizadores. Com uma interface intuitiva e fácil de usar, os viajantes podem realizar todas as etapas necessárias antes de sua viagem, desde a reserva do bilhete até o check-in e o acesso a informações atualizadas sobre seus voos em um único lugar.

Inicialmente, planeávamos incluir uma ampla integração com diversas companhias aéreas, oferecendo aos utilizadores uma seleção abrangente de opções de voos. No entanto, devido à complexidade e aos desafios associados a essa abordagem, decidimos focar em algumas companhias aéreas principais para o lançamento inicial do sistema, com planos de expansão para incluir mais companhias no futuro.

Este sistema diferencia-se de outros produtos similares pela sua abordagem integrada e centrada no utilizador. Ao consolidar todas as funcionalidades relacionadas a voos em uma única plataforma, "GateMate" oferece aos utilizadores uma experiência mais conveniente e eficiente, tornando a viagem aérea mais acessível e agradável.

## Personas and scenarios



**Nome:** Ana

**Idade:** 32 anos

**Profissão:** Executiva de vendas

Descrição: Ana é uma executiva de vendas que viaja com frequência a negócios. Ela trabalha para uma empresa de tecnologia e é responsável por visitar clientes em várias cidades do país. Ana tem uma agenda apertada e precisa coordenar as suas viagens de forma eficiente para maximizar o tempo disponível para as suas reuniões e compromissos profissionais.

**Cenário 1: Reserva de Viagem**

Ana tem uma reunião importante agendada noutra cidade na próxima semana. Ela precisa reservar um voo que se encaixe na sua agenda lotada. Ana acessa o "GateMate" no seu computador e pesquisa por voos disponíveis para o dia e horário desejados. Ela encontra uma opção conveniente e realiza a reserva em poucos cliques, garantindo o seu bilhete para a viagem.

**Cenário 2: Check-in Online**

No dia de sua viagem, Ana está ocupada finalizando algumas tarefas antes de sair para o aeroporto. Ela recebe um lembrete do "GateMate" no seu telemóvel para fazer o check-in online. Ana abre o website, completa o processo de check-in de forma rápida e simples, e recebe o seu cartão de embarque eletrônico instantaneamente.

**Cenário 3: Atualização de Status de Voo**

Durante a sua viagem de volta para casa, Ana enfrenta um atraso no aeroporto devido a condições meteorológicas adversas. Ela fica preocupada em perder sua conexão para casa. Usando o "GateMate", Ana verifica o status de seu voo e recebe atualizações em tempo real sobre o atraso e as opções disponíveis para reagendar sua conexão. Com essas informações, ela consegue planejar adequadamente sua viagem de volta para casa.



**Nome:** Pedro

**Idade:** 45 anos

**Profissão:** Empresário

Descrição: Pedro é um empresário que viaja com frequência a trabalho e lazer. Ele costuma utilizar o aeroporto de sua cidade como ponto de partida para as suas viagens. Como um viajante experiente, Pedro valoriza a pontualidade e a praticidade ao realizar suas viagens.

**Cenário 4: Verificação de Horários de Voos no Aeroporto**

Pedro está no aeroporto aguardando o embarque para sua próxima viagem de negócios. Enquanto aguarda, ele precisa verificar o horário de seu voo para garantir que não perca a partida. Pedro dirige-se aos quadros de informações no terminal do aeroporto, onde pode visualizar os horários de partida e de embarque dos voos. Ele localiza o seu voo na lista e confirma os horários. Satisfeito com a confirmação, Pedro aproveita o tempo restante para revisar os seus planos de viagem e preparar-se para embarcar.

## Project epics and priorities

**Iteração 1: Funcionalidades Básicas e Reserva de Viagem**

* **Épica 1: Funcionalidades Básicas do Sistema**
  + Objetivo: Estabelecer as funcionalidades básicas do sistema para permitir a interação dos utilizadores.
  + Funcionalidades:
    - Criação de contas de utilizador.
    - Login e autenticação.
    - Interface de pesquisa de voos.

**Iteração 2: Check-in Online e Atualização de Status de Voo**

* **Épica 2: Check-in Online**
  + Objetivo: Implementar a funcionalidade de check-in online para os utilizadores.
  + Funcionalidades:
    - Desenvolvimento da interface de check-in online.
    - Integração com as companhias aéreas para emissão de bilhetes de embarque eletrônicos.
* **Épica 3: Atualização de Status de Voo em Tempo Real**
  + Objetivo: Fornecer aos utilizadores informações em tempo real sobre o status dos seus voos.
  + Funcionalidades:
    - Integração com sistemas de informações de voos para obter atualizações em tempo real.
    - Desenvolvimento da interface para visualização do status de voo.

**Iteração 3: Verificação de Horários de Voos no Aeroporto**

* **Épica 4: Informações de Voos no Aeroporto**
  + Objetivo: Facilitar a verificação de horários de voos no aeroporto para os usuários.
  + Funcionalidades:
    - Desenvolvimento de uma interface para visualização de horários de voos no terminal do aeroporto.
    - Integração com os sistemas de informações de voos do aeroporto para obter dados em tempo real.

# Domain model

<which information concepts will be managed in this domain? How are they related?>

<use a logical model (UML classes) to explain the concepts of the domain and their attributes>

# Architecture notebook

## Key requirements and constrains

<**Identify issues that will drive the choices for the architecture** such as: Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance issues? Does it need to be robust for long-term maintenance?

Identify critical issues that must be addressed by the architecture, such as: Are there hardware dependencies that should be isolated from the rest of the system? Does the system need to function efficiently under unusual conditions? Are there integrations with external systems? Is the system to be offered in different user-interfacing platforms (web, mobile devices, big screens,…)?

E.g.: (the references cited in [XX ] would be hypothetical links to previous specification documents/deliverables )

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

* The existing legacy Course Catalog System at Wylie College must be accessed to retrieve all course information for the current semester. The C-Registration System must support the data formats and DBMS of the legacy Course Catalog System [E2].
* The existing legacy Billing System at Wylie College must be interfaced with to support billing of students. This interface is defined in the Course Billing Interface Specification [E1].
* All student, professor, and Registrar functionality must be available from both local campus PCs and remote PCs with internet dial up connections.
* The C-Registration System must ensure complete protection of data from unauthorized access. All remote accesses are subject to user identification and password control.
* The C-Registration System will be implemented as a client-server system. The client portion resides on PCs and the server portion must operate on the Wylie College UNIX Server. [E2]
* All performance and loading requirements, as stipulated in the Vision Document [E2] and the Supplementary Specification [15], must be taken into consideration as the architecture is being developed.>

## Architecture view

Uma imagem com captura de ecrã, diagrama, Retângulo, Esquema

Descrição gerada automaticamente

A arquitetura de sistema proposta é uma estrutura altamente modular e escalável, baseada em microserviços e facilitada por um reverse proxy. Essa abordagem permite uma gestão eficiente de várias funcionalidades do sistema, garantindo flexibilidade, desempenho e facilidade de manutenção.

No núcleo dessa arquitetura, encontramos os microserviços, unidades autônomas e especializadas que executam tarefas específicas. O primeiro entre esses é o **User CRUD**, responsável por todas as operações relacionadas aos utilizadores. Desde o registo inicial até a gestão de perfil e autenticação, esse microserviço interage diretamente com a base de dados de utilizadores, fornecendo uma interface consistente e segura para manipulação de dados de utilizador.

Em paralelo, o **Flights CRUD** assume o controle das operações relacionadas aos voos. Gerenciando uma base de dados dedicada, este microserviço não apenas fornece funcionalidades básicas de CRUD (Create, Read, Update, Delete), mas também integra-se, via **message queue**, de forma inteligente com o **Live Data Receiver**. Esta integração é vital para manter informações de voo atualizadas em tempo real, permitindo que os usuários obtenham dados precisos e atualizados sobre suas viagens.

Outro componente crucial é o microserviço **Notification**, responsável por manter os utilizadores informados sobre eventos relevantes. Ao receber atualizações do Live Data Receiver via **message queue**, este microserviço envia notificações oportunas e personalizadas, garantindo uma experiência de utilizador envolvente e informativa.

Por último, mas não menos importante, está o microserviço **Payment**. Esta componente gerência o aspeto financeiro do sistema, tratando de transações de pagamento de forma segura e eficiente. Integrado com um serviço externo de pagamento e com sua própria base de dados dedicada, este microserviço assegura uma gestão financeira robusta e confiável.

Para facilitar a comunicação entre esses componentes e a interface do utilizador, entra em cena o **Reverse Proxy**. Este intermediário inteligente encaminha solicitações dos clientes para os microserviços apropriados, garantindo um roteamento eficiente e um balanceamento de carga adequado. Ao proporcionar uma camada de abstração entre a UI e os microserviços, o reverse proxy simplifica a arquitetura geral e promove a escalabilidade e a disponibilidade do sistema.

## Deployment architecture

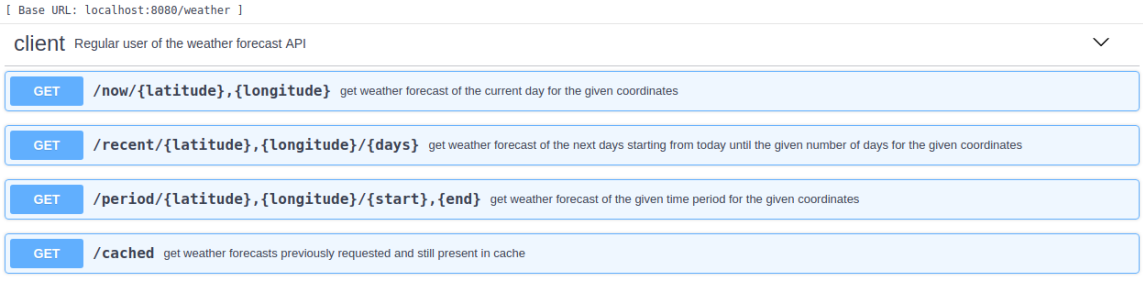
[Explicar a organização prevista da solução em termos configuração de produção (*deployment*). Anotar, no diagrama, as tecnologias de implementação, e.g.: colo aro simbolo do PostgreSQL na Base de dados,…]

# API for developers

[Explicar a organização da API. Os detalhes detalhes/documentação dos métodos devem ficar numa solução *hosted* de documentação de APIs, como o [Swagger](https://swagger.io), Postman documentation, ou incluída no próprio desenvolvimento (e.g.: maven site)

<what services/resources can a developer obtain from your REST-API?>

<document the support endpoints>



# References and resources

<document the key components (e.g.: libraries, web services) or key references (e.g.: blog post) used that were really helpful and certainly would help other students pursuing a similar work>